Foundations of an Artificial Neocortex

Florian Neukart[1], Sorin-Aurel Moraru[2]

[*1,2]Faculty of Electrical Engineering and Computer Science, University of Brasov, 500036 Brasov, Romania

[*1]florian.neukart@edu.campus02.at; [2]smoraru@unitbv.ro

Abstract

"Consciousness" is one of these suitcase words from psychology that helps us to discuss a complex subject no scientist or philosopher has been able to describe or demystify for millennia. It enables us to include yet unknown processes and (changes of) states associated with the human brain in our everyday-language with ease, not only without being able to describe what accounts for a conscious experience on neuronal or (sub-) atomic layers, but also without being able to explain what consciousness is on a more abstract layer. Lots of scientists from different fields have been working on disclosing the secret of consciousness, and numerous different explanations have been published and discussed controversially. Most of these theories feature a common denominator – the inclusion of a feature set, which is associated with the perception of consciousness. It would be counterproductive to reject such approaches, as only the detailed description and combination of single features will allow us to reproduce conscious behaviour in artificial entities. However, lots of these theories share one more thing: they try to simplify the enormous complexity of different information processing levels, such as self-conscious reflection, (self-) reflective or deliberative thinking (see also Minsky, 2006), or subjectivity in conscious experiences, which, amongst others, incorporates learned behaviour and experience. Simplification has been very applicable for physical theories, but we assume that more complex theories are required to approach the explanation of which features, brain states or processes consciousness comprises. We will not provide such a theory here, as this would go beyond the scope of this elaboration, but we will approach such one by starting to combine and technically explain features that we consider such a theory needs to take into account (Neukart, 2014).

Keywords

Artificial Consciousness; Quantum Computer Science; Computational Neuroscience; Computational Intelligence; Data Mining; Artificial Intelligence

Introduction

The description of quantum artificial neural networks [17, 18] has shown that in theory networks with arbitrary depth, such as the human brain is, can be created and processed. It is obvious that not only one artificial neural network (ANN) alone will be capable of processing all information provided by sensory inputs, such as vision. Thus, we consider that the following structures are required for achieving what we are looking for:

- Pattern recognizers
- Context classifiers (which are, generally speaking, pattern recognizers as well)
- Short-time memory
- Clustering ANNs
- Predictive ANNs

Additionally, access knowledge is required, thus to

- (knowledge) databases,
- ontologies,
- search engines,
- expert systems, and
    - information processors, thus agents
    - collecting
    - processing,
    - ranking, and
    - selecting context-sensitive information.

We not only suggest classical means of information technology and artificial intelligence, such as databases containing information or expert systems correctly accessing context-sensitive information, but also mentioned quantum artificial neural networks beforehand. It cannot be taken for granted that an artificial mind can only be implemented by processing information on a quantum computer (another approach is pursued by the human brain project), but clearly it would provide numerous advantages, such as processing power, e.g. in terms of arbitrary ANN depth. To date, only theoretical descriptions of quantum ANNs have been made, as one of the big challenges in quantum computer science still is the isolation of quantum physical systems from their own irrelevant properties, but that is just an aside – the resolution of this will not be scope of this elaboration. Even if our brain does not work in that manner, so that e.g. for learning or the execution of some limb motion information access and processing may happen on the classical neural level only, and experiencing conscious

content may involve signal processing on the quantum physical layer as well, a combination for an ACE seems to be promising. We will start the explanation of some ideas by discussing two further concepts, namely the one of context recognition as well as the concept of hierarchies. This is, because one of the major powers of our brains is that we can extract context from a situation. For this and other paradigms, such as language processing, to work, it is required to understand hierarchical information structuring and processing. The first concept mainly concerns the acquisition of knowledge and is discussed in the following part, whereas the second concept has to do with the organization of patterns in our (and the artificial entity's) brain, which is discussed at Implementation. Natural language processing (and understanding) is one of the key concepts for the human brain development, and therefore we will explain the proposed artificial neocortex based on the implementation of language processing.

Context Recognition and Hierarchical Learning

First of all, it is required to define what context actually means. Our brain is capable abstracting information, thus it can convert a situation rich on information into something more common, by implementing an inductive thought process and leaving off details. If we e.g. take a photograph picturing a laughing man and a woman, sitting in a meadow on a blanket under a tree, between them two glasses of wine and some food, then the context is that they are picnicking. A lot of information that may be depicted on the picture is irrelevant for a human brain correctly recognizing the context, e.g. it does not matter what exactly is on the blanket, or which clothes both wear. What is obvious for us is difficult to implement so that an artificial conscious entity is able to conclude the same. Before dealing with visual information, we will try to solve the problem with textual information such as speech or texts describing a situation. In terms of textual information, an example for a context problem is as follows: Let assume that there are three textual propositions:

1. Injury
2. Car
3. Speed

Let further assume that there exist two contexts:

a) Accident
b) No accident

Textual proposition 1 and 2 are true in the context of a, whereby propositions 2 and 3 are true in the context of b. Without further information, this is not only a complex situation for an artificially intelligent software system, but also for humans. The meaning of context permits the reconciliation of statistical measures of context examination utilizing the terms of context formalization, and for a series of propositions there exists a collection of sets of contexts, and the outer context C is defined by

$$true\big(C, \bigcap_{i=1}^{m} true(C_{ij}, T_i)\big) \forall j \quad (1)$$

where $T_1, \ldots, T_m$ represents as a series of textual propositions, and in case $\forall T_i$ there is a set of contexts C_{ij}. For each ithen true(C_{ij}, T_i) $\forall j$, which states that P_i is true for each C_{ij} , and C_{ij} are not structured hierarchically in advance, instead such are created according to specific sets of textual propositions. The number of existing contexts is assumed to be finite and to satisfy

$$C, C_{ij} \subseteq U_c \quad (2)$$

where U_c is the unity of all existing contexts. This results in two sub-problems:

1. What are the possible contexts C_i satisfying true(C_i, T) $\forall i$ if T is defined as single text?
2. If $T_1, \ldots, T_m$ is set of textual propositions satisfying the condition that for each text T_i there is a set of contexts C_{ij} so that true(C_{ij}, T) $\forall i$. What is the outer context C so that

$$true\big(C, \bigcap_{i=1}^{m} true(C_{ij}, P_i)\big) \forall j \quad (3)$$

The first problem requires some text T to be received, whereby each of the texts may feature a set of contexts that are true for T. An example is the phrase 'to put away', which can represent different activities, such as 'scoffing', or 'tidying away'. The challenge is how to figure out these possible contexts from all existing contexts.The second problem goes one step further in the sense that more than one text is presented, namely a series $T_1, \ldots, T_m$. Each of these texts T_i features a number of contexts and the challenge is to select the correct combined contexts of the remaining contexts for each T_i. The most suitable contexts are chosen by ranking in the stage of specialization and refinement. Considering the first given example, there exist the text propositions 'injury', 'car', and 'speed' with the two existing contexts 'accident' and 'no accident', whereby the textual propositions 'injury' and 'car' are true in context 'accident', while the propositions 'car' and 'speed' are true in context 'no accident'. We can

see that the outer context would include both 'accident' and 'no accident', which a contradiction is resulting from a lack of information. More information would result in higher contextual ranking and another outcome [2].

Definition of Context-sensitive Information

A textual context description consists of phrases, words or sentences, each of these describing accurately one facet of the full context. First of all, it is required to consolidate the data, which may be delivered in different forms, such as sentences, words or phrases. A simple procedure then is responsible for splitting the text into single words, which are then compared with the content of some dictionaries. The first (exclusion-) dictionary may be built upon words that do not help to understand the context of a situation, such as personal pronouns or articles. As a next step, it would be required to classify the remaining words, thus to translate them into knowledge. For this, additional domain-specific dictionaries are required, the number depending on what knowledge the artificial entity should be able to process. We have to bear in mind that also humans do not feature domain-spanning knowledge, so we should not presuppose that an artificial entity can do that at first. However, in theory there are not bounds. Each single word is then compared with the contents of the knowledge dictionaries and will be considered for further context analysis, if it is found. A short algorithm describing the situation is as described in table 1:

TABLE 1 WORD COMPARISON

Start
1. Collect and split text
2. Create context-array A_c and exclusion-array A_e
3. ***For each*** word
   a) compare word with exclusion-dictionary
      If word matches with entry in exclusion-dictionary updateA_e *with word*
      Else update A_c with word
4. ***For each*** entry in A_c
   a) compare word with knowledge-dictionaries
      If word matches with entry in knowledge-dictionaries do nothing
      Else remove word from A_c

End

Breakdown:

A_c: Array list holding the words used for context-analysis

A_c: Array list holding the words ignored at context-analysis

Basically, this can be compared with how context-sensitive information in brains is pre-processed. Information useless for exactly interpreting what happens in a situation is removed – this has been mentioned beforehand as abstraction.

Information Clustering

The textual information gathered in the last step will be used to query one or many information clusters, such as our brain would do. When considering the example of two picnicking persons again, then our brain would not search for information regarding the situation in our memories of cars or houses. Instead, our hierarchical thinking starts with conceiving the overall information on the picture to specific information depending on memories, knowledge and experience. This is what we also have to do with the memory of the artificial entity, such as a context-database. A context-database may be implemented by a distributed file system oriented database, such as Apache Hadoop [3], allowing the storage and management of hundreds of millions of files. However, before querying specific files, thus knowledge, defined by specific contexts, it is required to get back to self-organizing feature maps. As our human brain, this specific kind of artificial neural network is capable of clustering content. Furthermore, human thinking happens hierarchically, which will also be considered in the introduced approach.As an artificial entity should also be able to learn new things and to extend its knowledge, continuous clustering of information is absolutely required. Otherwise, each context interpretation would happen over the entity's overall knowledge, which would result in incorrect interpretation with high probability. Therefore, it is required to automatically detect and create preliminary contexts on existing knowledge (which we have to differ from preliminary contexts one level deeper, of which more lately). The approach applied is a tree view based hierarchical document clustering approach [4]. As the SOFM is an unsupervised learning ANN, no prior training is required, which makes it a perfect approach for clustering a huge amount of unstructured data, which the knowledge of the entity in consideration without doubt should be. Before being able to cluster the information, some pre-processing must be applied, such as indexing by the

vector space model (VSM) [5] for determining the occurrence frequency of words or terms within a document, resulting in a juxtaposition of the term occurrence frequency in documents. Again, a so-called stop-list or exclusion-dictionary is applied on all the documents contained within the distributed file system database for discarding information with little knowledge and importance for context interpretation, followed by the application of stemming algorithms [6]. The VSM-term-document-matrix forms the basis for further processing, such as described in *Context analysis*.

Context Analysis

After only the context-sensitive textual information has remained, it has to be compared with information stored in context-databases. As already mentioned, a context-sensitive database may be implemented by a distributed file system oriented database, such as Apache Hadoop, in the sense that stored contextual information refers to clusters of files. This context-sensitive database is then queried with the textual information determined within Definition of context-sensitive information, whereby each query delivers the foundation for contextual information; the delivered information can only form a foundation, as most of the files will not contain only information useful for correctly interpreting knowledge. Instead, the information within the search results needs to be clustered, resulting in preliminary contextual information. Clustering can be done by done by an algorithm such as the Term Frequency / Inverse Document Frequency (TF/IDF) [7], which is used in information retrieval for assessing the relevance of terms in documents from a document collection.

Hierarchical Learning

The hierarchical representation of information is a method that is strongly applied by our brains, so it has proved to be a valuable approach. Hierarchical clustering via SOFMs works in the same way as non-hierarchical clustering would do, with the difference that the underlying document structure is segmented into different levels, which are then processed subsequently (with still one SOFM). Furthermore, it is required to apply an algorithm for the growth of the SOFM, as the optimal structure cannot be determined in advance, preferably one that allows the ANN to take arbitrary structures. Algorithm capable of this is are the growing-SOFM algorithms[8,9], first growing the structure of the SOFM to the optimal size, and finally fine-tuning it. An advanced algorithm suitable for our purposes is as follows, whereby the number of starting nodes may vary:

TABLE 2 GROWING SOFM

Start

1. Initialize weight vectors of starting nodes randomly
2. Calculate growth threshold T_g for the given data set according to definition
3. Present input to network
4. ***Repeat***

   For each

   input value

   i. Determine the winning neuron n_w

   ii. Apply weight vector adaptation to neighbourhood of winner n_w and winner itself only by

   $$w_{j(t)} = \begin{cases} w_{j(t-1)}, & if\ j \notin N_t \\ w_{j(t-1)} + \alpha_{(t-1)}\left(x_t - w_{j(t-1)}\right), & if\ j \in N_t \end{cases}$$

   iii. Increase winner's error value

   If

   $$TE_i < T_g$$

   If

   i is boundary node

   a. grow node

   b. Initialize new node weight vectors to match neighbouring node weight

   Else

   distribute weight to neighbours

   iv. Initialize learning rate to its starting value
5. ***Until node growth is reduced to a minimum level***
6. Reduce learning rate and fix a small starting neighbourhood
7. Find winner and adapt the weights of winner and neighbours in the same way as in growing phase.

End

Breakdown:

T_g: growth threshold

n_w: weight vector closest to input vector mapped to current SOFM

TE: total error of node i

α: learning rate

As the approach should be as dynamic as possible, it should take into consideration both the growth between already existing neurons and boundary neurons, on the contrary to commonly applied SOFM growth algorithms. The growing grid algorithm [10] extends an existing structure by inserting new neurons as layers (rows or columns) within existing nodes, whereby as the basic growing SOFM algorithm inserts new neurons at the boundaries only; however, in the former approach the weights are determined by interpolation, as well as in the latter for the insertion between existing neurons. The weights for inserted neurons at the boundaries are then determined by extrapolation. As both hierarchical and growing features need to be combined, an approach relying on

independently spun and dynamically growing one-dimensional SOFM [4], applying the Growing SOFM algorithm for growth seems to be the most useful for our purposes. The use of independent SOFMs has the advantage that hierarchies can be modelled perfectly well. For correctly parsing the clusters, thus finding the correct branch of knowledge within the context-related search so that not the whole knowledge must be parsed, it is required to label these in a suitable manner, e.g. by LabelSOM[11], which is capable of determining the features of the input space that are most relevant for the mapping of an input vector onto a specific unit. Basically, this happens by determining the contribution of every element in the vector towards the overall Euclidean distance between an input vector and the winners' weight vector, which forms the basis of the SOM training process[11]. At first, the quantization error (QE) for each term within a cluster has to be determined (the beforehand mentioned Euclidean distance between word vectors and weights). The QE is then used for performing a ranking, whereby the first n words represent the label of the cluster or hierarchy node. A neuron's weight as well as the document similarity may be determined by

$$d(a,b) = \sum_i |a_i - b_i| \quad (4)$$

which is the Manhattan-metric, where $d(a,b)$ is the distance between two points a and b, and i is the current position in the two-dimensional coordinate system. The number of starting nodes for two-dimensional SOFMs within this structure adaptation is usually four, which may remain the same for one-dimensional maps, with the difference of one-dimensionality. The minimum number would be two. After the top-level clusters have been detected, clustering with one-dimensional SOFMs below these clusters can be done, so that a hierarchical structure of SOFMs would be the result.

Interpreting the Context

Now that the artificial entity commands over context-sensitive knowledge this knowledge can be applied on incoming interpretation requests. What a hypothetical artificial entity would be capable of so far is

- Abstraction
- The reduction of information to relevant parts
- Hierarchical knowledge structure
- The creation of a hierarchical knowledge structure based / creation of preliminary contexts on knowledge provided to the entity

This is quite a lot, but before being able to show intelligent behaviour of any kind, be it extraverted by speech or deeds, it must be able to connect the abstracted information with its knowledge. For this, the whole knowledge base is queried with the context-sensitive information determined from the input (*Definition of context-sensitive information*). The result is then one to many different contexts, based on the hierarchical learning the artificial entity has conducted. For each context, the number of appearances of the words is compared and based on this, a ranking can be created. Furthermore, a lexical database such as WordNet[12] should be applied, so synonyms are included in the search, as well as the lexical meaning of words.

TABLE 3 CONTEXT INTERPRETATION

Start

1. Receive pre-processed input texts
2. ***For each*** Term
   a) Query database
      i. Determine full meaning by inclusion of lexical databases
      ii. Apply algorithm for word comparison on lexical meanings
      iii. Query database with pre-processed input texts, lexical meanings and synonyms
      iv. Determine contexts
   b) Summarize similar contexts
   c) Determine
      i. number of actual text appearances
      ii. number of actual document references (if the documents have been linked)
3. Rank contexts based on number of text appearances and number of document references
   Calculate context weighting[12]
      i. Determine difference between each value of the number of references and its nearest lower value neighbour d_a
      ii. Determine difference between each value of the number of appearances and its nearest lower value neighbour d_r

$$W_c = \sqrt{\left(\frac{2d_a r_m}{3a_m}\right)^2 + {d_r}^2}$$

4. Determine maximum weight value
5. Determine all contexts appearing before the maximum weight value

End

Breakdown:

r_m: maximum number of references

a_m: maximum number of appearances

d_a: appearance difference to nearest lower neighbour

d_r: reference difference to nearest lower neighbour

It is not necessarily required to perform a ranking

according to the weight equation; however, results seem to be better than when just performing a ranking based on the number of appearances only.

Hidden Markov Models and Conceptual Hierarchies in the Neocortex

Apart from artificial neural networks, there exist lots of other classifying, predicting, auto-associating and clustering (network) structures. One that we need to discuss here is a special model developed by the Russian mathematician Andrei AndrejewitschMarkow, the Markov chain, or more precisely, the hierarchically hidden Markov model, derived from the hidden Markov model, which in turn has been derived from simpler Markov models such as the Markov chain. Not all processes always run into a fixed deterministic order and are therefore relatively easy to describe. Usually a system follows basic rules though, but it may not behave deterministically – we consider our brain to be such a system. Also, there are processes that cannot be observed directly but from which we can detect and evaluate signals. These must then be sufficient to draw conclusions on the true events. For a description of such hidden running, nondeterministic processes to statistical modelling, such as the hidden Markov models, may be used. Hidden Markov models are now with various issues, such as speech and handwriting recognition, in Biology for classification of protein sequences, as well as for modelling of economic processes.Hidden Markov models (HMM) indicate a two-stage stochastic process. The first stage corresponds to a Markov chain whose states are not visible from outside (hidden). A second stage generates a stochastic process with so-called observations, which are the output symbols that can be observed at any time according to a probability distribution. The aim is to conclude from the sequence of output symbols upon the sequence of the non-visible states. HMMs are used in the recognition of patterns in sequential data, e.g. stored speech sequences in the context of speech recognition or price behaviour in the stock market. The hidden states of the Markov chain correspond to semantic units that are that have to be detected in sequential data, which are semantic models. A hidden Markov model describes a two-stage stochastic process. According to their capabilities, hidden Markov models may be applied for three types of problem statements, which are

- Decoding

  A model M and an observed sequence of Sis given. Within the decoding problem it is determined what the most likely path through M that generates S is, which may happen according to the Viterbi algorithm.

- Classification

  A model M and an observed sequence Sis given. Within the classification problem it is determined how large the total probability $P_M(S)$ of M is, so that S is emitted, which may happen according to the forward algorithm.

- Training

  A set of training sequences and a model structure is given. Within the training problem it is determined what the best model parameters (state transition- and emission probabilities) that the training set allows are, which may be done according to the Baum-Welch algorithm.

Hidden Markov models are important in the task of engineering an artificial mind, as they allow abstraction, which we already mentioned is a major strength of our neocortex. Artificial neural networks do that as well, however, we do not consider it as the correct approach to solely rely on one paradigm. What is more, hierarchically hidden Markov models as auto associative pattern recognizers have proven to outperform ANNs in some domains, such as speech recognition. Furthermore, HHMMs work hierarchically, such as the pattern recognizers in our neocortex, activating only the higher hierarchy-pattern recognizers required for further processing incoming data. Thus, we will continue examining HHMMs.HHMMs can be seen as hierarchical and recursive generalization of the hidden Markov models discussed beforehand. The difference to the former ones is that each of the hidden states becomes a probabilistic model as well, which simply means that each of the hidden states is a hierarchically hidden Markov model as well. This is important, as the structure of the neocortex is organized in that way as well – neural structures are organized in lists whose entries are lists again, or in other words, pattern recognizers of a lower conceptual hierarchy refer to pattern recognizers on a higher hierarchical level (see *Implementation*). Hierarchically hidden Markov models thus do not emit single symbols, such as described beforehand, but sequences. HHMMs extend the already discussed HMMs in the sense that they are not restricted to emitting single observations. The states of HHMMs are HHMMs themselves, which again

contain sub-states and which are capable of emitting strings of observations. States being able to emit strings of observations are called abstract states, on the contrary to states that have single emissions and are called production states. The internal states of HHMMs may be called recursively, so that after the run has been completed control is returned to the abstract state. HHMMs can be represented by HMMs, where each HMM state consists of the production state and the abstract state higher up the hierarchy of the HHMM[13]. However, it should be mentioned that both the HMM and the HHMM belong to the same type of classifiers and may be used for solving the identical set of problem statements. Basically, the HHMM may be transformed into a standard HMM; however, the latter one utilizes its structure for solving a subset of those problem statements more efficiently.

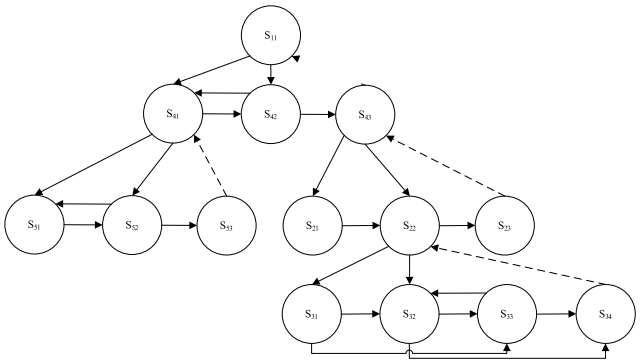

FIG.1 - HIERARCHICALLY HIDDEN MARKOV MODEL

The point when a state in an HHMM is activated it will activate its own particular probabilistic model. Thus, it will activate one of the states of the underlying HHMM, which then may activate its underlying HHMM and so forth. The procedure is rehashed until the production state is activated, which we learned are the states emitting observation symbols in the HMM sense. The point when the production state has emitted a symbol, control comes back to the state that activated the production state. The enactment of a state in a HHMM under an internal state is known as a vertical transition. After a vertical transition is finished, a horizontal transition occurs at a state inside the same level. The point when a horizontal transition results in a terminating state, control is come back to the state in the HHMM, higher up in the hierarchy that generated the last vertical transition. A vertical transition can bring about additional vertical transitions before arriving at a succession of production states and at last coming back to the top level. Consequently, the production states visited offers ascent to a succession of observation symbols that is processed by the state at the top level. The strategies for evaluating the HHMM parameters and model structure are more intricate than for the HMM.

Implementation

We already got to know that the neocortex is organized in a hierarchical structure. Additionally, we got to know some of the ingredients required for implementing an artificial neocortex, where the most important ones are

- different sorts of (quantum) pattern recognizers, comprising
  - auto associative,
  - classifying, and
  - predictive ones
- learning algorithms for pattern recognizers
- self-organization and artificial self-organizing structures
- differently structured databases, and
- search engines.

The neocortex consists of 6 layers, each of which is connected with the next-higher and/ or next-lower layer. Within these layers, it is assumed that hierarchical information processing occurs, which means that at first information from a lower conceptual layer is passed over to the next-higher conceptual layer in the sense of 'rough' pattern recognition up to detailed pattern recognition. This can be imagined very simply by the recognition of a face. A face consists of numerous features, but at first it is a face. Thus, the lowest conceptual layer is responsible for detecting basic structures, maybe rough shapes. There is not only one pattern recognizer active at once, but all. FIG.2 - Pattern presented to multiple pattern recognizers shows the presentation of a pattern to two pattern recognizers – in fact it would be hundreds of thousands or more. This is, because it is required to present the basic face-patterns to all pattern recognizers at once – this is one of the massive parallel processing tasks we often hear of in the context of the human brain.

After the input has been presented, some of the pattern recognizers will fire, if the pattern has been recognized. What happens if it is a new pattern, such as a face we have not known beforehand, will be discussed later. So, assuming that the pattern has already been learned, some of the pattern recognizers will fire and transmit a signal via their axon to the next

higher hierarchy, but not to all pattern recognizers within this level, but rather to specific ones, namely the ones the pattern recognizers of the first level are connected to. The signal they transmit to the next-higher hierarchy tells the pattern recognizers in the higher hierarchy that they have to expect a pattern that they should try to recognize. The pattern is then passed over to all of the pattern recognizers of the next-higher hierarchy, which then do the same as the pattern recognizers on the next-lower level. The pattern is then passed over up to the highest conceptual level, where it is finally recognized. FIG.3 - Hierarchical pattern processing shows a simple auto associative ANN (left) passing over its result to another pattern recognizer one hierarchy above (right).

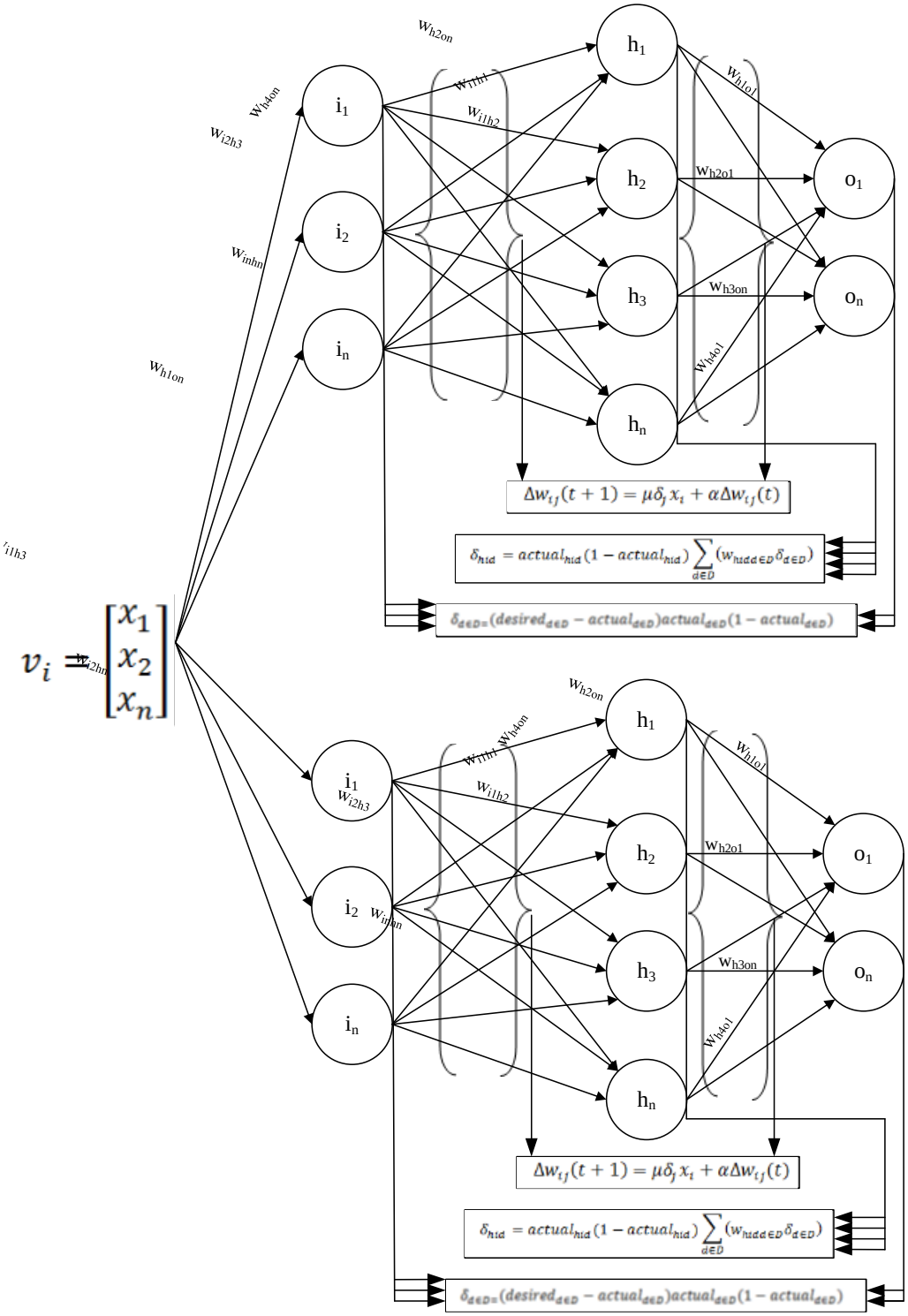

FIG.2 - PATTERN PRESENTED TO MULTIPLE PATTERN RECOGNIZERS

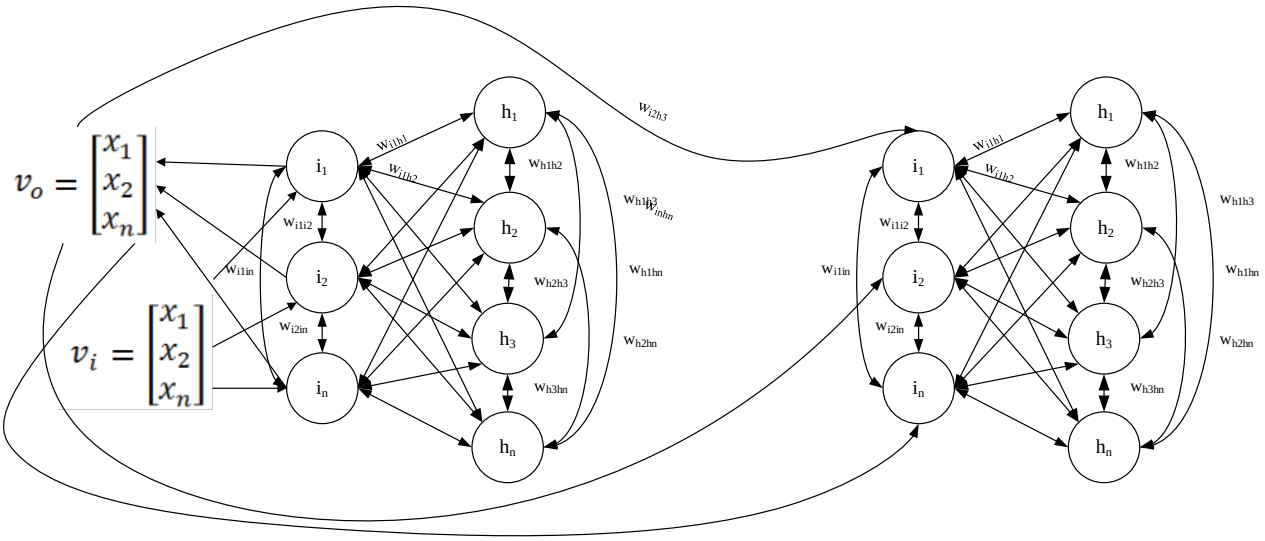

FIG.3 - HIERARCHICAL PATTERN PROCESSING

There are thousands of pattern recognizers active at once, as we need to consider that not only all pattern recognizers on the lowest conceptual level need to be presented the incoming pattern, but also that each stored pattern (such as a specific face) is represented by numerous pattern recognizers, as a face must be recognized also when it is distorted, like when it is wearing sunglasses or a beard, or when it is viewed from the profile instead of from the front, or when the light is lighter, darker, … This is basically what our neocortex consists of (and what allows us to philosophize about the human mind or to read and think about this text):

- Specific pattern recognizers, and their
- redundant counterparts, as well as
- hierarchical structures of pattern recognizers.

However, hierarchies do not only work in one direction, it is also assumed that pattern recognizers on higher hierarchical levels inform pattern recognizers on lower conceptual levels that a pattern is likely to be recognized. Thus, we have to deal with communication in both ways, up the hierarchy when lower level-pattern recognizers recognize a pattern, and down the hierarchy when higher-level pattern recognizers assume that a pattern will likely be recognized. This happens, as higher-level pattern recognizers do receive input from numerous lower-level pattern recognizers, and when a cluster of lower level-pattern recognizers is not able to auto associate the pattern they are responsible for (such as a mouth that is not only distorted by e.g. a beard, but covered by a blue scarf), some others responsible for the same higher level-pattern (such as a face) may recognize the pattern they are responsible for (such as eyes or a nose). Triggered by these lower level-pattern recognizers, the respective higher level ones may have enough information so that the overall pattern (such as the face of a specific person) can be recognized and pass over the information to the lower-level cluster that has failed to recognize the pattern. The information that could not be identified (the mouth with the scarf) is then learned, thus stored into a new pattern recognizer for this specific face, which we can easily create within an artificial entity. It is also possible that the specific face is linked to the pattern of the blue scarf, which our entity may have already been stored, so that the face of the specific person is also recognized easily in the future if it is partially covered with a blue scarf. In our biological neocortex (and most likely also in an artificial one), it is not always granted that a distorted pattern is recognized correctly. Most of us have made this experience when we fail to recognize a specific person we know very well or if we think that we have recognized a specific person that actually is someone that we do not know. From a computer science point of view, the hierarchies can be represented as multi-dimensional arrays of pattern recognizers, or lists pattern recognizers referring to other lists of pattern recognizers. We got to know two methods of implementing pattern recognizers, which are artificial neural networks and (hierarchically hidden) Markov models. Both have their qualities, thus both with find application in the proposed solution.

Acquisition of Basic Knowledge

First and foremost, it is required that the artificial

entity is capable of learning. However, learning is a very loose concept, thus it has been split in four major parts, which overlap. Besides, the sequence of the steps discussed here is not predefined – some of them may be carried out in parallel, some of them in sequence. To begin with, it is required that the artificial entity features knowledge, but how can access to knowledge be given? There exist several different approaches, whereby we consider access to the internet as the first and foremost thing. Furthermore, different knowledge databases should be provided, which may be very different in structure, thus file-system oriented, relational or even OLAP-databases. The internet contains most of the knowledge of our species, and we are used to access it for learning new things. This is what an artificial conscious entity should also be capable of, however, access to the internet alone is not where acquisition of knowledge can start, as the ACE must be able to ask questions, as we do it. Thus, what is required at first is the capability of asking questions, or better, purposeful search for information. As an approach, we suggest to use a file system with information deposited into knowledge categories, such as discussed in the chapter*Context recognition and hierarchical learning*. This may be implemented by a file system oriented database as well as meta information referencing to the correct knowledge categories. This information is not stored hierarchically, thus in categories and subcategories. For the ACE being able to access and extend knowledge purposeful, it is required to structure this knowledge so that it is stored hierarchically, which can be done by a self-organizing clustering method, such as a self-organizing feature map, or even better by an approach incorporating adaptive resonance theory ANNs, such as fuzzy ARTMAP. Approaches relying on ART do not suffer from the stability/plasticity-problem, which means that also the input data can be self-organizing or dynamic and change over time. Thus, anew clustering would not require complete re-training of the clustering approach, as continuous training allows the network to keep old knowledge as well as adapt to new. After the knowledge has been clustered, access may be given to limited knowledge databases, such as Wikipedia (we call it limited, because although Wikipedia contains millions of pages belonging to most of the knowledge categories we can differentiate from each other, the whole internet contains even more). Another step required is that the ACE is allowed to cross-reference the knowledge learned, which has been partially done by the permission for assigning knowledge to not only one, but many categories, such as an apple may be accessible through the category 'plants' as well as the category 'food'. Furthermore, access to synonym databases such as WordNet is required, so that the ACE is not only able to access information, but also synonyms, which can be used to further interconnect information. Moreover, such databases provide access to verbs and adjectives belonging to specific nouns as well, so that even more information interconnection may be achieved. For the extension of existing knowledge categories different document similarity measurement methods may be applied, a simple one being TD/IDF already discussed at *Information Clustering*. This implies that only a specific amount of knowledge for each category is stored, which we suggest should happen both in distributed file system databases, as well as via meta information, such as links to websites containing more knowledge. We suggest this shared approach for the following reasons:

- For the ACE it is not useful to store and organize the whole internet as a distributed file system database, although technically viable. The internet as knowledge entity is dynamic and subjected to continuous changes, thus it would require enormous resources for continuously querying, storing and clustering newly available information.
- It is not required for the ACE to have all existing knowledge to its avail in its brain (which the distributed file system database is a part of), as it is possible to store meta information, such as links, additionally to specific knowledge on a high conceptual level, as well as to let the ACE query the internet as we do it when we do not know something.
- As a next step, it is required to encode the conceptual and hierarchically structured knowledge into patterns.

Additionally to the textual information, pictures to specific kinds of knowledge should be stored and linked to the respective textual concept, such as the pictures of apples to the textual concept of an apple. Thus, the ACE now not only has textual, but also visual information to its avail.

Encoding the Acquired Knowledge into Pattern Recognizers

We humans do not store knowledge in the form of texts within our brains, but within patterns. One may

argue that if the ACE accesses its knowledge by a query of its databases, and puts out an answer to a question that has been based on a statistical weighting, then this does not go along with understanding. However, this is the same what happens within our brain when we access knowledge. Beforehand we mentioned that it has been estimated that within our neocortex there exist around $3 * 10^8$ pattern recognizers, which are organized hierarchically. This means that some pattern recognizers on the lowest conceptual level are able to recognize, say, the crossbar of the letter A. On higher conceptual levels the whole letter is recognized. If the brain is dealing with a word, such as 'Apple', then the same happens with all the letters within the word, so all of the letters are recognized and signals from the pattern recognizers of the lower levels converge on a higher conceptual level – the word 'Apple'. However, this is not quite right, as we are dealing with auto associative neural networks, which are able to complete the word if some parts are missing, if the pattern as a whole is disturbed in a not overly great manner, or even during the pattern is being processed. Let assume, the first four letters of the word have been read and processed, then the 'Apple'-pattern recognizers are already in standby position and may also communicate down the hierarchy to the 'e'-recognizers telling it that it as a whole may be required to fire soon. What happens then is that the firing thresholds of the 'e'-recognizers are lowered so that they can fire more easily. However, we have been writing about pattern recognition and the querying of clustered and hierarchically stored textual information. Again, we consider a shared approach as useful, which means that knowledge within a database for querying textual information is absolutely useful, but it is also required to translate visual information into queries, and this requires auto associative pattern recognizers. Such auto associative pattern recognizers, often thermal artificial neural networks, are presented specific patterns, such as the pattern of an apple, and then they are able to recall it, even if it is slightly modified or disturbed. For a simple concept such as an apple (and all other concepts as well), it is required to create hundreds of pattern recognizers with different inputs, and to structure the pattern recognizers hierarchically. The human brain is not organized highly redundant, as simply not enough neurons are available for achieving this. Thus, the input pattern must be split up in simpler patterns at first, or better, in different input streams directing their information directly to all pattern recognizers on the lowest conceptual level. One of the input streams may present the curves of the apple's round shape, one its colour, one its muster … The human visual cortex processes 12 such input streams, and what happens is the same. The lowest conceptual pattern recognizers (all of them) are addressed, and some of them fire. Thus, if one of the recognizers responsible for curved shapes, one for the colour red, one for specific muster of an apple, and some more fire, they fire up the hierarchy to all of the pattern recognizers they are connected to. All other lower-hierarchy pattern recognizers do nothing, as no pattern has been recognizes (as edges, e.g.). What happens is that in the best case only one of the apple-pattern recognizers has received inputs from most of the lower-hierarchy pattern recognizers that fired. Certainly, this does not only happen via two hierarchies – the brain e.g. uses six, based on the fact that the neocortex is six-fold, whereby this means that there exist six layers of abstraction, each of which is organized hierarchically.Possible means for implementing a hierarchical artificial neural structure are so-called deep learning architectures, such as Hinton's deep belief network or the deep belief-like network. Basically, such complex networks consist of stacked ANNs, such as restricted Boltzmann machines, where the output of each layer (each Boltzmann machine) serves as input for the next layer. The problem is that such architecture only make use of some sort of hierarchical processing for preparing the initialization of weights, which is not exactly what is required here. What would be required is that each of these auto-associative layers is trained separately on specific patterns, resulting in a structure in which lower level pattern recognizers activate processing in higher level areas. However, there is another challenge: hierarchical processing within brains happens does not only pass over the input from one lower level pattern recognizer to higher hierarchies, but from many. So if we again imagine the image of the apple (which is not stored as an image in our brain, but as a hierarchy of patterns), some shape pattern recognizers, some colour-recognizers and more do fire at once and propagate their signals via axons up to the higher hierarchical level. Again, on the next level, some pattern recognizers fire as they have been activated from the next-lower level by the patterns of the lower level (we remember that the ANNs used are auto-associative, so what they do is to reproduce a pattern when they recognize it). Processing in that way continues until the highest conceptual hierarchy has

been reached and finally, the apple has been recognized. Thus, information from lower conceptual hierarchies is converged in higher conceptual hierarchies, so the chosen approach should combine ideas from decision trees and ANNs. A decision tree is a structure consisting of nodes organized in a tree structure, whereby there are two kinds of nodes:

- Internal nodes, which make local decisions based on the local information they have to their avail.
- Terminal nodes, which are used for making a final decision.

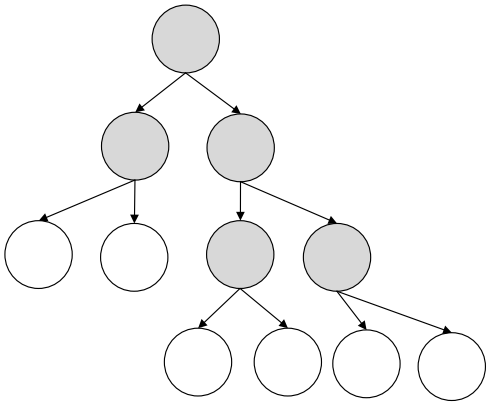

FIG.4 - BINARY DECISION TREE

FIG.4 - Binary decision tree shows a tree where each of the grey internal nodes results in two decisions (thus binary). The terminal nodes are represented by the white nodes. Each internal node makes its local decision based on a function f(x), whereby a binary decision tree may simply decide according to

$$d_i = \begin{cases} left, & if\ f(x) < 0 \\ right, & if\ f(x) > 0 \end{cases} \quad (5)$$

and usually

$$f(x) = x_i - a_i \quad (6)$$

wherea_i represents the feature a on the i^{th} node that is used for coming to local decisions. The tree assigns a distribution of data to each terminal node, which is then used for coming to a final decision. If the decision tree is not binary, then it is required to build classes, so data belonging to the class c_1, is structured beneath the terminal node n_1. The weather outlook, e.g., can be structured in classes, such as 'rainy', 'sunny' or 'cloudy'. Thus, if all data assigned to a node belongs to the same class, the node is a terminal node, as no further decisions than the final decision can be made. If this is not the case and more than one class exists, a decision function selecting one of the nodes has to be

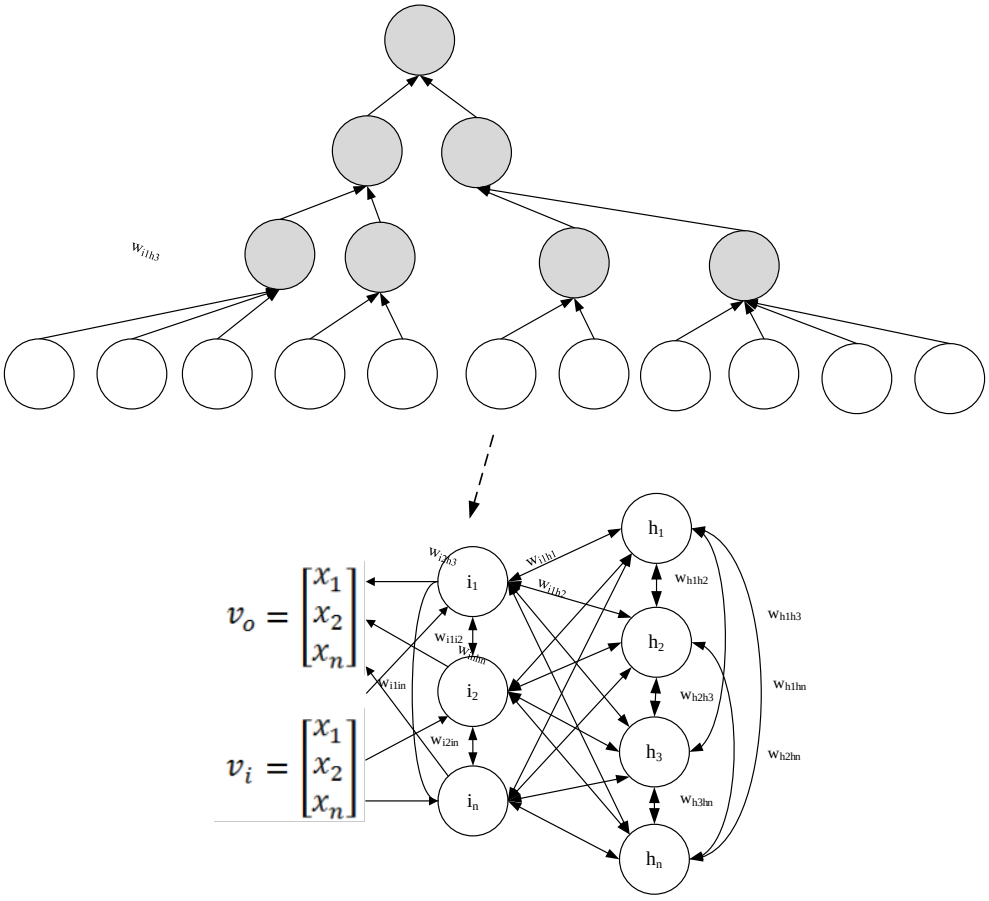

FIG.5 - BOTTOM-UP ANN TREE

defined. Although decision trees are easy to define and understand, they may become very complex for complex problem statements, and hierarchical pattern recognition in human brains for sure is. If it was the size only, this can be tackled by multivariate decision functions, such as the linear combination of the features. In case of hierarchical information processing, the decision tree must be created and processed bottom-up, and each of the nodes is represented by an auto-associative ANN (see FIG.5 - Bottom-up ANN tree):

The next difference is that we are not exactly dealing with a decision tree, but with a structure that is like a decision tree. Would the ANN tree be processed top-down, then the result would not be the activation of just one terminal node, but of many. Furthermore, due to redundant use of patterns, lower level pattern recognizers do not just propagate their patterns up the hierarchy in a decision tree-like manner, as each node may have more than one parent node (see FIG.6 - Cumulative activation).

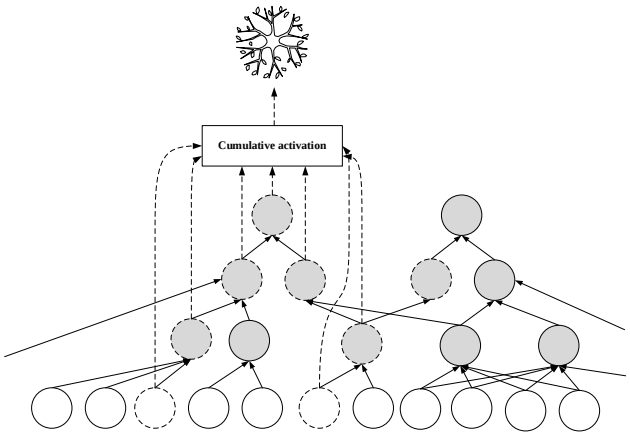

FIG.6 - CUMULATIVE ACTIVATION

The dashed lines show the activation of some lower level pattern recognizers, which in turn activate some higher level pattern recognizers and so on. In FIG.6 - Cumulative activation, the cumulative activation of all dashed pattern recognizers results in the recognition of (a specific sort of) tree. Furthermore, on the third hierarchy from below on the right side a pattern recognizer is activated by the firing of a pattern recognizer from below as well, however, in the example this does not result the activation of the highest level pattern recognizer (for another kind of tree) – cumulative activation of more nodes from the right path would be required. Such structures can be implemented easily, as well as pruning and adding of connections between the nodes. What has to be considered is the number of input neurons for each pattern recognizer, whereby the optimal number of input neurons does not necessarily need to be defined in advance for optimally recognizing a specific pattern – auto associative ANNs are even capable of recognizing their specific patterns when these are incomplete or disturbed.

Access to Knowledge

Now that the knowledge has been structured and visual information has been encoded into patterns, these two components have to be set into relation. Although within a human brain everything is stored in patterns, even all knowledge of different domains a person may possess, we do not consider it required to do the same for an ACE. For the internal information processing of an ACE we consider it required to distinguish between

- knowledge about world and
- expert knowledge.

There exist many more subcategories and we are aware that especially the second category provides an ample scope, but in the beginning it is essential that the creators of the ACE distinguish between them. After the ACE has been equipped with basic knowledge about the world and some expert knowledge, it can learn and take over categorization itself. The first category incorporates visual information about the environment the ACE is moving, which includes basic information about entities, objects and their purpose, as well as context-sensitive information, so far as possible. Getting back to the example of the picnic, visual information about entities and objects may include patterns of people, flora, fauna and food that the ACE is capable of processing visual information in a way that allows it to correctly recognize what it perceives. Contextual information is modelled in patterns as well, as the activation of the visually perceived objects does not only allow the identification of such, but also the recognition of a situation, such as the picnic. We do not consider it as a requirement that a context such as a picnic is recognized at first, as even we fail in some situations and the ACE is capable of learning.The second category incorporates knowledge of the world as well, such as descriptions of objects on atomic layer or specific knowledge of information technology. Such expert-knowledge does not necessarily need to be encoded into patterns as within our brain, but may be stored in large and distributed database structures, comprising different database types such as

distributed file system-, OLAP- and relational databases, whereby the first category may serve as storage for knowledge stored in pages or files such information provided by Wikipedia, and the latter ones e.g. for processing complex (knowledge-related) stored procedures, for storing the results of calculations, or even for the analysis of specific knowledge stored in a cube.Databases may also serve as short-term extension to a short-term memory that has been modelled in patterns, but more on that later. The connection between knowledge and patterns is enabled by translating incoming visual information (patterns) into textual information. If one is of the opinion that the translation of visual information into text for accessing knowledge is nothing a biological conscious entity would do, we admit that she is correct; however, we do not see an obstacle in creating an ACE by changing the way of how information is accessed. Thus, the suggestion is to store simple and shot textual information additionally to each pattern (such as the word 'apple' for the pictures of apples), which then can be used to query the knowledge stored in databases when patterns are presented through visual channels.

Language Processing and Understanding

Clustered, interconnected knowledge provided via textual and visual information does not enable the ACE to communicate, so the ability to process natural language or another approach for communication is required as well. Language processing incorporates numerous different approaches and paradigms, such as probabilistic models of language, statistical natural language processing, information extraction, text mining, robust textual interference, statistical parsing, grammar induction, constraint-based theories of grammar, and computational lexicography. An approach that can be applied are hidden Markov models (see *Hidden Markov models and conceptual hierarchies in the neocortex*). Such models have proven to perform well at processing speech or parts of speech. Part-of-speech tagging (POST) is the part of corpus linguistics that will serve for explanations here. Basically, POST considers the context as well as the definition of a particular part of speech for making up a word within a text, such as a document or speech, as belonging to a specific part of speech. Part of speech refers to a linguistic category of words, generally defined by thesyntacticormorphologicalbehaviour of the lexical item in question. POST is more difficult than just considering words and their correct part of speech, as words may have more than one meaning. An example is the German word 'einen', which can both be an indefinite article and a verb. A commonly used example in English is 'The sailor dogs the hatch', where the word dogs alone may represent the plural of dog, what would be definitely wrong in that context, in which it is used as a verb. Both grammatical and semantic analysis may result in the desired tagging, where via the former one analysis would state that the plural noun is definitely wrong here, and the latter one that the verb would occur in the context of sailor and hatch more likely. Summing up, POST is the task of tagging each word in a text with its appropriate part of speech, such as

The[AT] representative[NN] put[VBD] chairs[NNS] on[IN] the[AT] table[NN].

or

The[AT] representative[JJ] put[NN] chairs[VBZ] on[IN] the[AT] table[NN].[14]1

A well-known POST-corpus is the Brown-corpus[15], from which the tags above came from. The first example shows that the word 'representative' has been tagged as a singular noun, whereby in the second example it has been tagged as an adjective. The words 'put' and 'chair' allow for the same. Thus, some sort of probability model has to be applied for correctly identifying the context of a sentence, and one of them are HMMs. It has already been explained that processing of the current state within HMMs only depends from the last step. The states are represented by POST tags, whereby observations are sequences of words. The transition probability is represented by the bigram model for POST tags, the observation probability by the probability of generating each token from a given PIST tag. A bigram is every sequence of two adjacent elements in a string of tokens, which can be e.g. words or letters. The bigram frequency distribution within strings can be used for statistical analysis of text, such as within HMMs:

$$P(W_n|W_{n-1}) = \frac{P(W_{n-1},W_n)}{P(W_{n-1})} \quad (7)$$

where the probability P of a token W_n with the preceding token W_{n-1} is given by the probability of their bigram, which is the co-occurrence of the two tokens $P(W_n|W_{n-1})$ divided by the probability of the preceding token. Again, hidden within the HMM means that the sequence of POS tags (states) that is

[1]Dror Gideon (2009): Part-of-Speech Tagging [2013-12-08]; URL: http://www2.mta.ac.il/~gideon/courses/nlp/slides/chap10_pos.pdf

responsible for generating the sequence of words (observations) is hidden. The Viterbi algorithm is used within HMMs for decoding, which we remember is the discovery of a best hidden state sequence Q given an observation sequence O (and an HMM). The Viterbi-algorithm therefore determines the most likely sequence of hidden states in a given HMM and an observable sequence of symbols. Having very shortly discussed the HMM decoding problem, which asks to discover the best hidden state sequence Q by a given observation sequence O and an HMM $\lambda = (A, B)$, we now know how λ can determine POS-tags for words. However, before an HMM can do that, it must be trained, similar to an ANN which also has to be trained. There exist three types of training for HMMs, which are supervised, unsupervised and semi-supervised. In supervised training all training sequences are tagged, meaning that e.g. a human expert has labelled the words with tags beforehand. The model thus would be trained to learn that 'Students' is a noun, 'need' is a verb, etc. whereas in unsupervised learning all training sets are unlabelled, thus the tags are unknown.

The forward–backward algorithm is used for finding the most likely state for any point in time, on the contrary to the Viterbi-algorithm, which is used for finding the most likely sequence of states. Assuming an HMM with N states (or POS-tags, for language processing), its initial parameters are set randomly. Until the HMM converges, the forward-backward algorithm is used for determining the probability of various possible state sequences for generating the training data (often described by E(xpectation)). In the Baum-Welch algorithm, these probabilities are then used for re-estimating the values for all of the parameters of the HMM (often described by M(aximization)). Each iteration modifies the HMM-parameters in a way that guarantees for the increase of the likelihood of the data ($P(O|\lambda)$). The algorithm can be stopped at any time before convergence for getting an approximated solution.That leaves us at a point where POST of words becomes possible, which forms one of the basis features of natural language processing and in consequence, natural language understanding. These two fields comprise numerous subfields of research that has to be dealt with, on which numerous books and other sorts of publications have been written. For us, it was important to describe a paradigm that an artificial entity can make use of for processing and even understanding language, such as the hidden Markov model, and how information processing within such a model works. Furthermore, in *Hidden Markov models and conceptual hierarchies in the neocortex* similarities between the neocortex and HMMs have been discussed, so it is only fair to describe its basic functional principles here.

Quantum Pattern Recognizers

In [1], abig partwas dedicated to quantum mechanics and theoretically possible applications in machine learning [1, 18]. This has been done with a special purpose, namely the discussion of a possible processing approach for the numerous pattern recognizers of the ACE. Even if it may not be that case that quantum effects are required for producing conscious experiences in human brains, we may nevertheless make use them within ACEs, if advantages prevail. Breaking down quantum information processing to single pattern recognizers, it has already been discussed that the training performance can be significantly increased by lifting all possible configurations of the ANN in quantum linear superposition at once and searching for the required configuration with Grover's search algorithm. However, it has been shown that with an ANN of arbitrary largeness the search for the solution on the performance register would still require exponential time with respect to the composite weight vector's size, which is

$$O\left(\sqrt{\frac{2^b}{T}}\right) \tag{8}$$

There are ways to deal with this situation, such as the application of randomized training algorithms, but there are further advantages that suggest the implementation of quantum neurocomputer, and this is the capability for solving so-called hard problems. The problem of combinatorial explosion could be solved by exploiting physical phenomena directly, such as analogue or quantum computers do. Initially, the idea was described by Richard Feynman[16] who was able to show that exponential complexity related to calculated probabilities can be reduced to a problem of polynomial complexity related to simulated probabilities, and a similar proceeding may be applied to NP-complete problems. A neurocomputer, which our brain or to some degree also an artificial neural network is (the latter one is not exactly a neurocomputer, as a classical ANN is not processed on neurons) belongs to another problem class, and compared to NP-hard problems, their complexity comprises not only an exponentially large number of

simple computations, but an unknown analytical structure. Unfortunately, an analogue computer is not universal, slow and inaccurate. A possible way for tackling the problem may be the introduction of non-deterministic approaches to computations, which may be the not here discussed Monte Carlo method, as well as randomized algorithms. Both can be utilized for solving combinatorial problems, and the theory of computational complexity states that polynomial time nondeterministic algorithms, compared to polynomial time deterministic ones, are more efficient in the sense that there exist algorithms of the former category that can solve problems probably within polynomial time or even within polynomial time for sure. To put it briefly, this means that such algorithms exchange complexity with completeness or correctness. Nevertheless they provide a solution for the problem, would not there be the problem of number randomization – random number generators are slow and unreliable in the sense that no real random numbers are generated; instead an algorithm is used for it. Quantum physics allows the creation of an analogue computer with real randomization, which means with such a device polynomial time nondeterministic algorithms can be directly implemented and solved on the hardware, and it is universal. Back to quantum artificial neural networks and the brain, very often we mentioned auto associative neural networks, which are trained to recognize one or more patterns even if the presented pattern is distorted or incomplete. The associative memory problem can be described as the storage of qn-dimensional patterns as dynamical attractor:

$$\xi_i^{\eta}(\eta = 1,2,\dots,q; i = 1,2,\dots,n) \quad (9)$$

and if a pattern ξ_i is then presented to the network, its similarity is compared to the stored patterns ξ_i^{η}. If the presented pattern is similar enough (depending from the allowed error rate) to one specific stored pattern, it is related to the specific attractor and a dynamic process is initialized, which eventually converges to the sample pattern. It has been explained in[17] that within a QANN the goal is to find the configuration of weights, or Hamiltonian H in quantum terminology, which provides a solution within the pre-defined ANN parameters, which in terms of an associative QANN are a prescribed number of attractors at specific locations and of specific type. Zak and William[18] state that compared to a classical ANN, the approach is completely different, as the structure converges to an attractor via a training algorithm. Arguments for favouring QANNs (instead of classical ANNs) are

- The dimension of the unitary matrix (H) implemented on quantum hardware may be exponentially larger within the same space. As a consequence, the QANN capacity, or the number of patterns the QANN is able to store, as well as the number of dimensions of these patterns, are exponentially larger as well.
- Through interference of patterns, a sort of grammar may be introduced to a QANN (apart from that, only a QANN with high dimensionality and complexity would be able to process such information). This means that if e.g. letters are stored in the form of stochastic attractors ξ_η are stored within the QANN then the simultaneous presentation of a specific number of letters to the QANN is accompanied by quantum interference effects in the sense that they will converge to a new attractor, which not only preserves the identity of the single letters, but simultaneously is not only a sum of these. This is similar to sentences consisting of words, and words consisting of letters. By changing the phases of the single components H_{ij} , also the grammar may be changed, let say from English to German.
- QANNs feature attractors representing stochastic processes, as described above.

The application QANNs within an ACE would therefore be beneficial, as a lot more complex and deep artificial neural networks could be used. The major problem with deep ANNs has always been their training, although some approaches have been found that deal with the problem. However, such algorithms either deal with the depth, thus the vertical structure of an ANN, or the horizontal structure, thus the number of neurons within a hidden layer. Additionally to restricted Boltzmann machines, we personally have made very good experience with genetic or thermal algorithms for training deep structures, especially as the latter work well independent from both structural dimensions. However, when dimensionality increases, also such algorithms' performance decreases exponentially. Thus, for training deep and broad networks such as the one(s) in a human brain, the search for the optimal configuration via Grover's algorithm, be it via entangled performance registers or in another way, is a lot more promising than attractor dynamics in classical ANNs.

Real World Input and New Experiences

Now that the ACE is capable of learning from different kinds of databases (whereby we consider the internet as heterogeneous and unstructured database here as well) and structuring this information efficiently, it is required to determine how sensual information is interpreted. Well, for with respect to visual information both spatial and temporal information is of relevance, thus how the content of a scene changes with respect to time. The information we visually perceive is much unstructured, but may be organized on a higher conceptual level. For example, when the ACE would walk down a street, heading from one location A to another location B, then it may interpret objects, which belong to one of the low conceptual levels of the scene (we call it scene, although the whole process may consist of several scenes) simply by scanning the environment. A low conceptual level in this scene may comprise cars, people, a street, a crossing. With this information, the ACE may gain contextual information, such as described in *Context recognition and hierarchical learning*. Contextual information in this scene may comprise information such as 'busy', 'accident at crossing', 'dangerous', 'how to pass crossing', etc… Equipped with this contextual information, behaviour can be derived, either from stored patterns or databases. Behaviour may be the avoidance of collision with objects or other people, how to pass a crossing, etc… we will not go into technical detail here on how visual information can be interpreted with pattern recognizers, as this has been already discussed extensively. However, the information described above could also have been extracted from a static image, thus does not necessarily feature a temporal component. It is improbable that all people and objects in our scene are static; instead they are moving and have their own targets. For the ACE, it is not required to know all these targets, as even we humans do not know that. Nevertheless, what we do is to predict the situation. Our neocortex has been developed for prediction and it does that all the time. It predicts where a car may be in 2 seconds and from that we derive if it is save to cross the street or not, or it predicts where someone heading towards us will be most likely in 3 seconds, so that we can avoid collision. This is also what an ACE should do, which means that it is on the one hand required to access knowledge or trained ANNs, but on the other hand to train such instantaneously for predicting patterns as quickly as possible. For this, not auto associative ANNs are used, but feed forward ones, or better, QANNs, as only the latter ones may be trained quickly enough for predicting a complete scene. For sure, it is not required to predict the whole scene, but in the given example to extract features of moving objects for a time sequence, and when one of these objects becomes relevant for the ACE, to predict its behaviour and derive own behaviour from the results.Not everything may be interpreted correctly, and an inexperienced ACE may make mistakes, as inexperienced humans do. Consider a child having never seen a butterfly before and thus running after one, ignoring its environment, which may not be dangerous on a flat meadow, but in a city. Thus, not even new information about objects, but also about context are collected continuously. For an ACE, there is no need to forget anything, even if the information is of no relevance for its desires and aims. Basically, the interpretation of new information happens in the same way as with basic knowledge, described at *Acquisition of basic knowledge*, with the difference that it is stored and brought into context with existing knowledge. This then transforms new information into knowledge, or experience. The difference between experience and knowledge is that experience is based on knowledge, thus it is a higher conceptual level of information. From a computer science point of view, experience may be interpreted as a cluster of knowledge, whereby knowledge here does not only comprise hard facts, but of stored situations linked to stored knowledge. These stored situations may be enriched with Meta information and tagged, so that they can be queried and compared to an actual situation, which is then handled based on the results, thus experience. If the handling was successful although the situation differed from the experience in a higher amount, then the related experience-cluster may be extended by the variational information of the new situation. Knowledge may be stored as new pattern recognizers (which encode patterns), and a situation may be stored in manifold ways, one of which we would suggest are sequences situation-relevant patterns.

Automatic Information Interconnection

A human brain does interconnect information, which is the reason why we can learn in one domain from another domain, or explain complex information such as physical processes in metaphors. This is what is also required to happen within an ACE. Information interconnection may be implemented in different areas and ways. The intercompatibility of patterns or a

cascade of patterns may e.g. be verified by understanding the type of a problem, or 'what it really is'. Having understood that evolution, the behaviour of ants when searching for food, the cooling of metal or learning within an artificial neural network all are optimization problems, has resulted in the creation of ANN training algorithms that copy paradigms from nature e.g. for predicting weather. The type of a problem can very often be figured out easily by understanding its nature, and understanding its nature has very often to do with understanding context. The implementation of checking for pattern intercompatibility is a meaningful means for solving unsolved problems, thus the ACE may be able to derive solutions from different categories of knowledge, as we humans do.

A Superior Goal

Now that the theoretical ACE features some interesting capabilities, it must be equipped with a superior goal. IBM Watson's goal e.g. was to answer Jeopardy questions, but this was not a superior goal it pursued itself. The goal of its ingenious developers was to show that Watson can understand and intelligently process input provided in natural language, which in fact was an extraordinary achievement. However, when Watson is not provided input, he is not pursuing a superior goal, such as continuously improving itself (in fact, he may partially do that in the sense of structuring and learning information, but only when he is told to do so), or contributing to the world's progress. Our goals, for example, are anything but easy to explain, from a scientific point of view (see also [20]).For all we do, be it the contact to others, eating or moving we have motivation, which is a central drive deeply rooted in our brain and established evolutionary. The clock for the motivation is the reward system that regulates what we perceive as pleasant or less pleasant by the release of the neurotransmitter dopamine. In former times, when our neocortex has not been as developed as it is now, dopamine may have been released by successfully hunting down an animal, and nowadays it may be the reward of our boss or the money we collect each month. For the ACE, if it should be included in our society, the same reward principle may be implemented – only the goals may be more flexible than we are able to define ours (a remark: books have been written on why and how we act as we do, thus only a very simplified approach will be explained here). This sounds difficult at first, but there is a machine learning approach where we actually apply this, namely particle swarm optimization, in which individuals of a swarm represent solutions to a problem moving through n-dimensional space. The movement of each individual is influenced by its local and best-known position as well as the best known positions in the whole search-space. These positions are updated as soon as better positions have been detected by other individuals, which results in an overall movement of the swarm towards the best solution. Schematically, the best solution in case of an ACE may be encoded as the superior goal, whereby the superior goal must be implemented as continuously moving towards infinity, so that it can never really be reached. This is not something mean, as if we think on contributions for improving the world it is far from being perfect – moreover, who can really describe what a perfect world for everybody is like?

Conclusion

From what has been discussed within this elaboration it becomes obvious that from our point of view it will soon be possible to create artificial conscious entities. The required technology and methodology already exists or is researched on in ambitious projects such as the human brain project [19]. The question therefore is not if it will be possible or not, but how this will influence the world we live in. Ethical questions must be prompted, as we will have to decide how we treat artificial conscious entities, namely as conscious beings or as objects such as a today's computer, and which rights will be granted to them, because if an ACE is capable of questioning the world and oneself it will also question its creators. Apart from that, if we go one step further and consider what it means for humanity and evolution if we advance technology in a way that allows to transfer our minds into an artificial vessel, not only questions of ethics, but also from religious groups will come up. We personally pursue a transhumanist opinion and consider the next step in human evolution towards machine consciousness is not an option, but an absolute requirement, as with continuous population growth the resources earth can provide us with will not suffice, and this will happen before we will be able to conduct interstellar travels to the nearest exoplanets.

REFERENCES

A. K. Heller et al. (2009): Infinite Hierarchical Hidden

Markov Models; Proceedings of the 12th International Conference on Artificial Intelligence and Statistics; Florida: Clearwater Beach

A. Rauber (1999): LabelSOM: On the labelling of selforganizing maps; Washington: Proceedings International Joint Conference on Neural Networks

A. Segev et al. (2007): Context recognition using internet as a knowledge base; Journal for Intelligent Information Systems; 29:305–327

Apache foundation: hadoop; [2013-10-29]; URL: http://hadoop.apache.org/

B. Fritzke (1995): Growing Grid: A self-organizing network with constant neighborhood range and adaptation strength; Neural Processing Letters, 2(5)

D. Alahakoon et al. (2000): Dynamic self organizing maps with controlled growth for knowledge discovery; IEEE Transactions on Neural Networks; vol. 11, pp. 601--614

F. Neukart (2012): On Quantum Computers and Artificial Neural Networks; Journal of Signal Publishing Research; PP.1-11

F. Neukart (2014): Reverse-Engineering the Mind [2014-02-17]; scribd. – the largest online library; URL: http://de.scribd.com/doc/195264056/Reverse-Engineering-the-Mind

G. Dror (2009): Part-of-Speech Tagging [2013-12-08]; URL: http://www2.mta.ac.il/~gideon/courses/nlp/slides/chap10_pos.pdf

G. Salton (1988): Automatic text processing: the transformation, analysis, and retrieval of information by Computer; Addison-Wesley: Massachusetts

G. Salton, M. J. McGill (1983): Introduction to modern information retrieval; New York: McGraw-Hill

H. U. Bauer, T. Villmann (1995): Growing a Hypercubical Output Space in a Self-Organizing Feature Map; ICSI Tech Rep. TR-95-030

Human brain project [2013-12-20]; URL: https://www.humanbrainproject.eu/de

Leeds University: Automatic Mapping AmongLexico-Grammatical Annotation Models [2013-12-08]; URL: http://www.comp.leeds.ac.uk/amalgam/tagsets/brown.html

M. F. Porter (1980): An algorithm for suffix stripping: Program, 14 no3, pp 130-137

M. Zak, C. P. William: Quantum Neural Nets; Center for Space Microelectronics Technology: Jet Propulsion Laboratory; Pasadena: Caltech

Minsky M. (2006): The Emotion Machine; Simon and Schuster: New York

R. Feynman (1982): International Journal of Theoretical Physics, Vol. 21, No. 6/7

R. Freeman et al. (2002): Self-Organising Maps for Tree View Based Hierarchical Document Clustering; Honolulu: Proceedings of the IEEE IJCNN'02; vol. 2, pp. 1906-1911

WordNet (2013): WordNet a lexical database for English [2013-11-03]; URL: http://wordnet.princeton.edu/